

DESARROLLO DE APLICACIONES SOBRE SCTP

Jorge López Aragonese

ITT: Telemática

Tutor: Carlos García Rubio



Índice

1. Introducción
 - Motivación y objetivos
 - Entorno de desarrollo
2. Estado del arte
 - Stream Control Transmission Protocol
 - Multihoming
 - API de sockets
3. Desarrollo de aplicaciones
 - Prueba 1: Cliente/Servidor en C
 - Streaming de video mediante VLC (API de sockets en C)
 - Prueba 2: Cliente/Servidor en Java
 - Streaming de video mediante JMF (API de sockets en Java)
4. Conclusiones y trabajos futuros
 - Conclusiones
 - Trabajos futuros
5. Bibliografía

1. Introducción

Motivación y objetivos

Motivación:

- SCTP → alternativa a TCP y UDP.
- Interesante para transmisión de datos en tiempo real.

Objetivos:

- Conocimiento del protocolo.
- Estudio del API de sockets.
- Pruebas de funcionalidad.
- Streaming de video.

Entorno de desarrollo

Compuesto por:

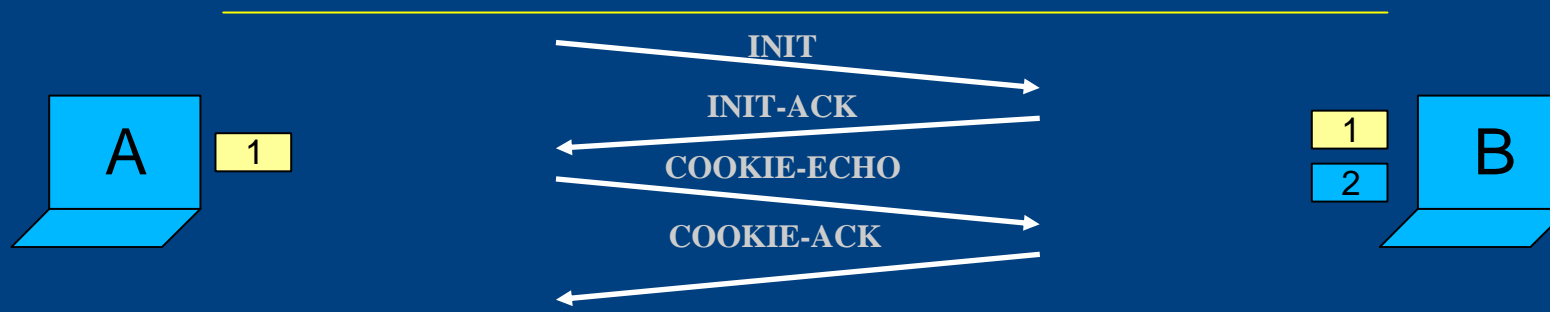
- 2 máquinas virtuales creadas con VMware.
 - Facilidad de creación de interfaces de red.
- Sistema Operativo SuSE Linux 10.2.
- Paquete lksctp para dar soporte a SCTP.

2. Estado del arte

Stream Control Transmission Protocol

- Definido en el año 2000 por el grupo SIGTRAN.
- Protocolo de nivel de transporte.
- Surge para transportar señalización telefónica SS7, aunque se pretende utilizar en otros campos.
- Orientado a conexión, confiable, control de flujo y secuenciación. Orientado a mensaje.
- Nuevo concepto de comunicación denominado asociación. Capacidad de realizar Multi-homing.

Multihoming

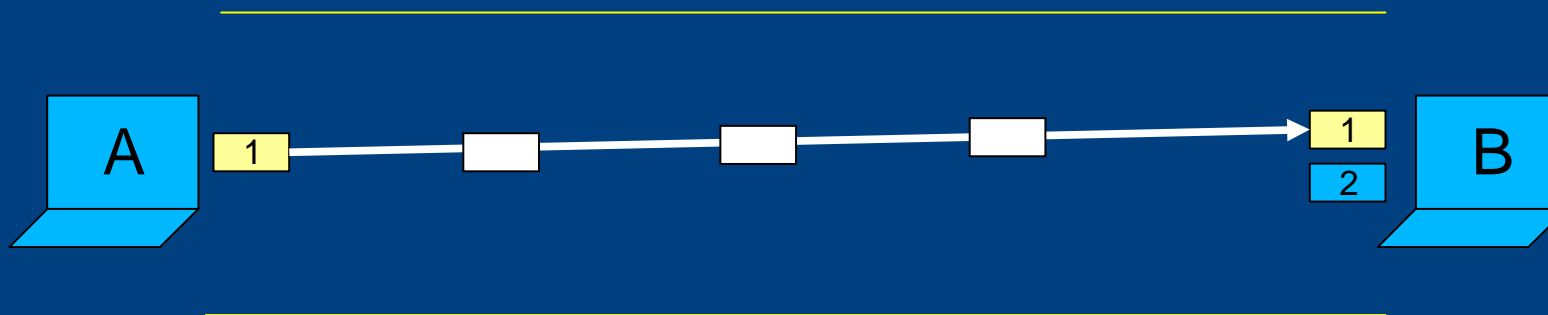


1) Creación de la asociación:

- Establecimiento de asociación en 4 pasos:

Se indican interfaces de red que formarán parte de la asociación.

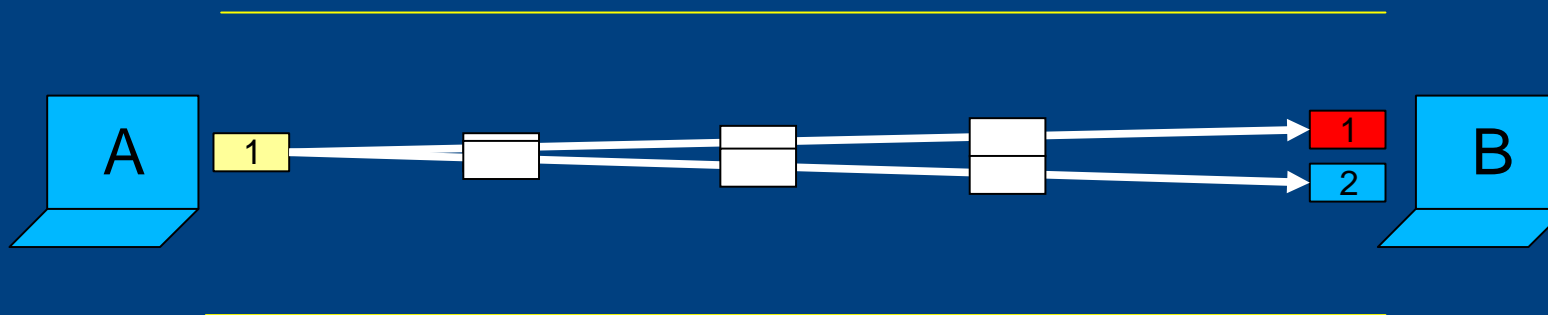
Multihoming



2) Flujo de datos:

- Realizado a través de los interfaces primarios.
- Concepto de Stream: 1 asociación \rightarrow N streams.

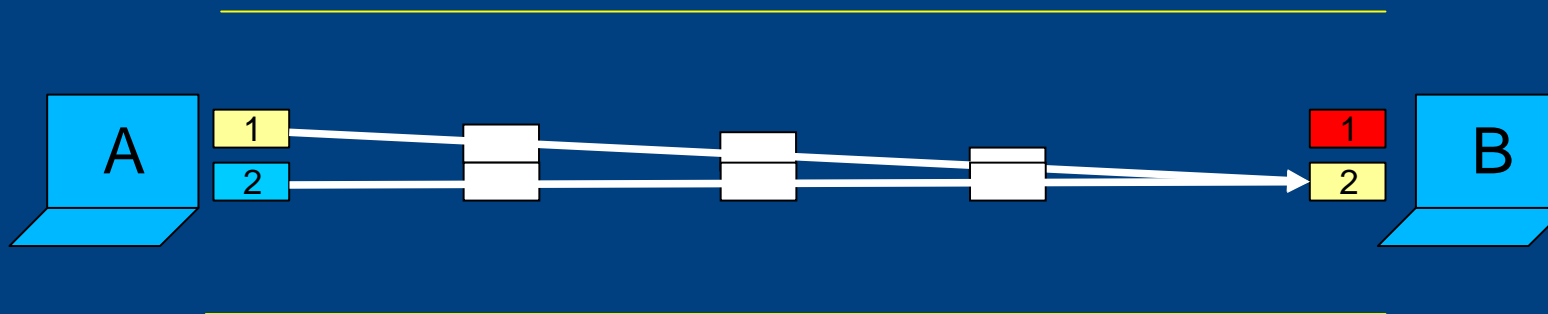
Multihoming



3) Fallo en socket B.1. 3 situaciones:

- Heartbeat, si hay tiempo de inactividad.
- B indica su nueva IP primaria a A: interfaz 2.
- Sobrepasado número determinado de reintentos.

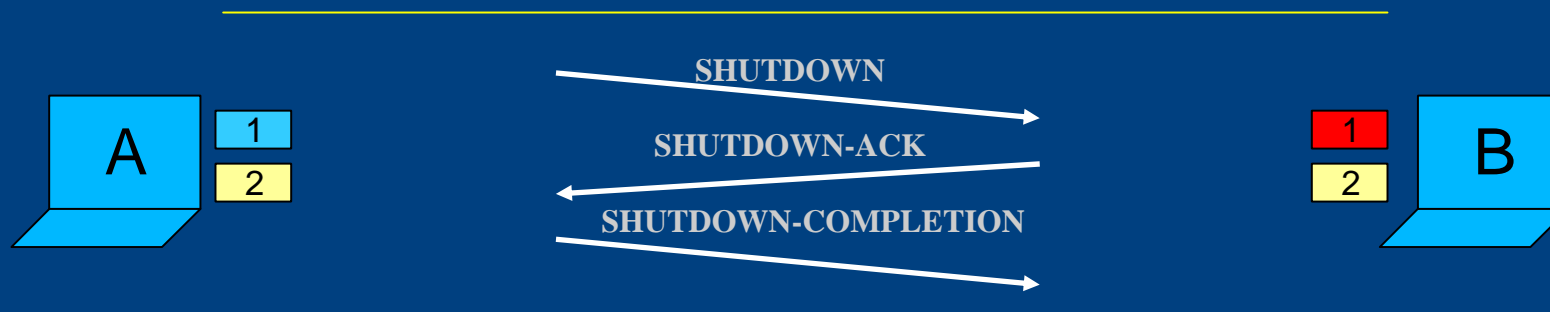
Multihoming



4) Capacidades del protocolo:

- Agregar interfaces de red en asociación ya establecida.
- Modificar dirección IP primaria de forma dinámica.

Multihoming



5) Finalización de la asociación:

- SCTP lo realiza en 3 pasos.
- TCP lo hace en 4. Estado de “medio cierre” poco utilizado.

API de sockets

- Se encuentra en fase de draft.
- Soporta todas las características interesantes para el desarrollo del proyecto.
- Permite utilizar dos tipos de socket:
 1. One-to-one style socket: 1 socket \rightarrow 1 asociación.
 2. One-to-many style socket: 1 socket \rightarrow N asociaciones.
 - Asociaciones diferenciadas por ID.
- Actualmente, sólo definido en lenguaje C.

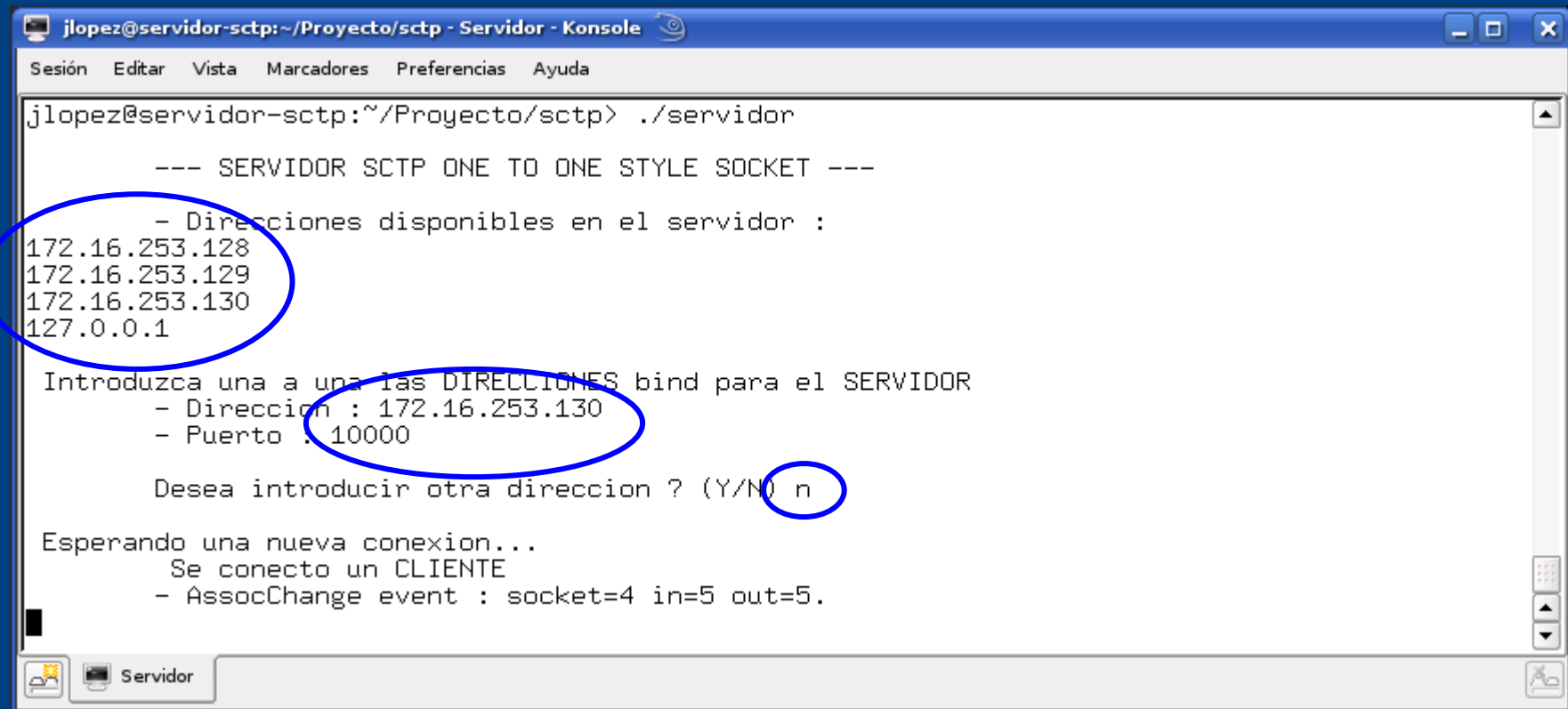
3. Desarrollo de aplicaciones

- Utilizando API de sockets en C:
 - Cliente/Servidor para probar API de sockets en C.
 - Streaming de video mediante SCTP.
- Utilizando API de sockets en Java:
 - Cliente/Servidor para probar API de sockets en Java.
 - Streaming de video mediante SCTP.

Prueba 1:

Cliente / Servidor en C

Cliente / Servidor en C



```
jlopez@servidor-sctp:~/Proyecto/sctp - Servidor - Konsole
Sesión  Editar  Vista  Marcadores  Preferencias  Ayuda

jlopez@servidor-sctp:~/Proyecto/sctp> ./servidor

--- SERVIDOR SCTP ONE TO ONE STYLE SOCKET ---

- Direcciones disponibles en el servidor :
172.16.253.128
172.16.253.129
172.16.253.130
127.0.0.1

Introduzca una a una las DIRECCIONES bind para el SERVIDOR
- Dirección : 172.16.253.130
- Puerto : 10000

Desea introducir otra direccion ? (Y/N) n

Esperando una nueva conexion...
Se conecta un CLIENTE
- AssocChange event : socket=4 in=5 out=5.
```

Cliente / Servidor en C

```
jlopez@servidor-sctp:~/Proyecto/sctp - Cliente - Konsole
Sesión  Editar  Vista  Marcadores  Preferencias  Ayuda

jlopez@servidor-sctp:~/Proyecto/sctp> ./cliente

--- CLIENTE SCTP ONE TO ONE STYLE SOCKET ---

Direcciones disponibles en el cliente
172.16.253.128
172.16.253.129
172.16.253.130
127.0.0.1

Introduzca una a una las DIRECCIONES bind para el CLIENTE
- Direccion : 172.16.253.128
- Puerto : 10001

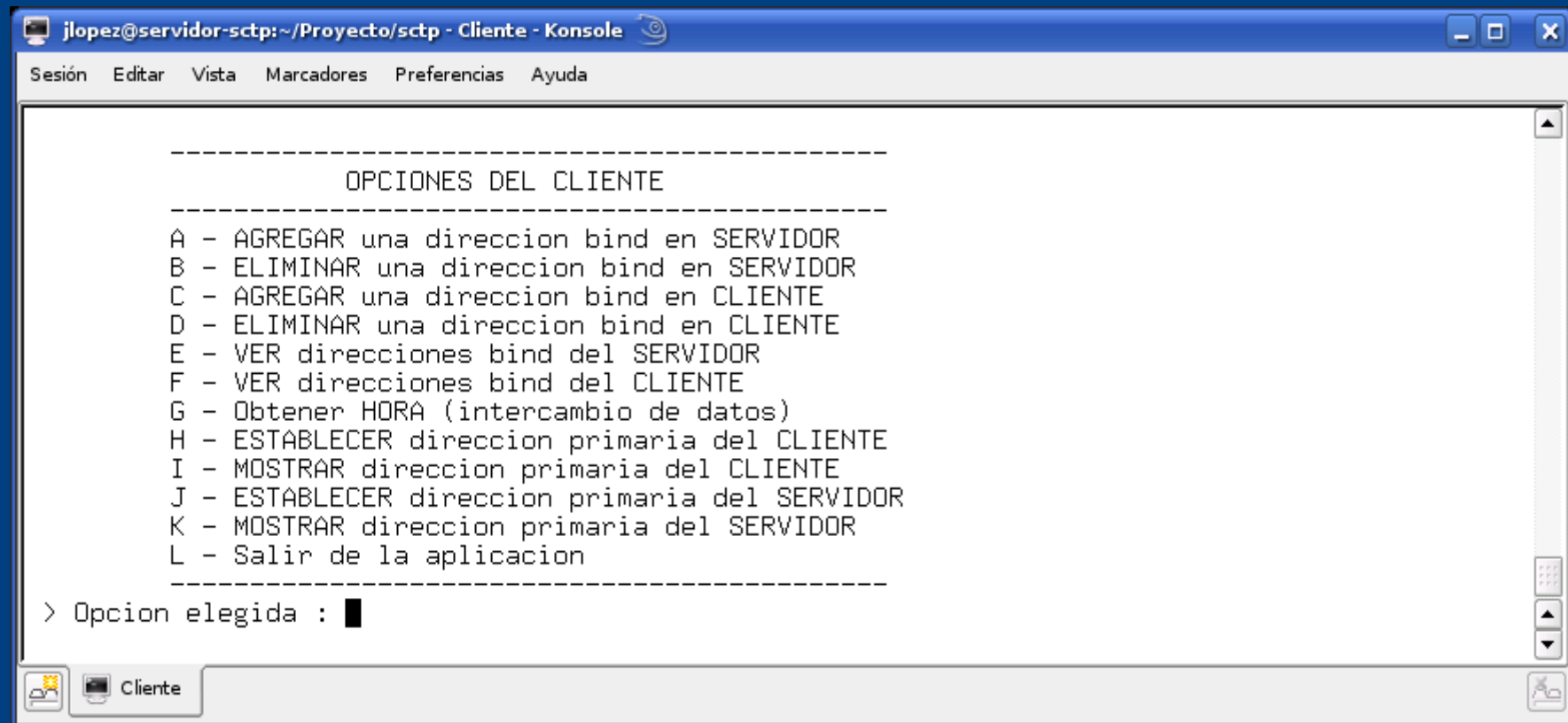
Desea introducir otra direccion ? (Y/N) n

Introduzca una a una las DIRECCIONES del SERVIDOR a las que desea conectarse
- Direccion : 172.16.253.130
- Puerto : 10000

Desea introducir otra direccion ? (Y/N) n

--> Se ha establecido correctamente la conexion con el servidor de hora <--
- assoc id   = 1
- state      = 4
- instrms    = 5
- outstrms   = 5
```

Cliente / Servidor en C



```
jlopez@servidor-sctp:~/Proyecto/sctp - Cliente - Konsole
Sesión  Editar  Vista  Marcadores  Preferencias  Ayuda

-----
                        OPCIONES DEL CLIENTE
-----
A - AGREGAR una direccion bind en SERVIDOR
B - ELIMINAR una direccion bind en SERVIDOR
C - AGREGAR una direccion bind en CLIENTE
D - ELIMINAR una direccion bind en CLIENTE
E - VER direcciones bind del SERVIDOR
F - VER direcciones bind del CLIENTE
G - Obtener HORA (intercambio de datos)
H - ESTABLECER direccion primaria del CLIENTE
I - MOSTRAR direccion primaria del CLIENTE
J - ESTABLECER direccion primaria del SERVIDOR
K - MOSTRAR direccion primaria del SERVIDOR
L - Salir de la aplicacion
-----
> Opcion elegida : █
```

Cliente / Servidor en C

No. ↓	Time	Source	Destination	Protocol	Info
1	0.000000	172.16.253.128	172.16.253.130	SCTP	INIT
2	0.000187	172.16.253.130	172.16.253.128	SCTP	INIT_ACK
3	0.000317	172.16.253.128	172.16.253.130	SCTP	COOKIE_ECHO
4	0.000451	172.16.253.130	172.16.253.128	SCTP	COOKIE_ACK
5	16.284854	172.16.253.128	172.16.253.130	SCTP	DATA
6	16.284940	172.16.253.130	172.16.253.128	SCTP	SACK
7	16.289776	172.16.253.130	172.16.253.128	SCTP	DATA
8	16.289844	172.16.253.128	172.16.253.130	SCTP	SACK
9	16.289973	172.16.253.130	172.16.253.128	SCTP	DATA
10	16.493019	172.16.253.128	172.16.253.130	SCTP	SACK
11	25.509389	172.16.253.128	172.16.253.130	SCTP	ASCONF
12	25.509486	172.16.253.130	172.16.253.128	SCTP	ASCONF_ACK
13	28.480076	172.16.253.130	172.16.253.129	SCTP	HEARTBEAT
14	28.480145	172.16.253.128	172.16.253.130	SCTP	HEARTBEAT_ACK
15	33.061789	172.16.253.129	172.16.253.130	SCTP	ASCONF
16	33.061869	172.16.253.130	172.16.253.129	SCTP	ASCONF_ACK
17	36.402695	172.16.253.129	172.16.253.130	SCTP	DATA
18	36.409763	172.16.253.130	172.16.253.129	SCTP	DATA
19	36.419595	172.16.253.130	172.16.253.129	SCTP	DATA
20	36.419664	172.16.253.129	172.16.253.130	SCTP	SACK
21	36.603679	172.16.253.130	172.16.253.129	SCTP	SACK
22	40.042564	172.16.253.129	172.16.253.130	SCTP	SHUTDOWN
23	40.043062	172.16.253.130	172.16.253.129	SCTP	SHUTDOWN_ACK
24	40.043139	172.16.253.129	172.16.253.130	SCTP	SHUTDOWN_COMPLETE

▾ ASCONF chunk

- ▶ Chunk type: ASCONF (193)
 - Chunk flags: 0x00
 - Chunk length: 32
 - Serial number: 0xdb7a6025
- ▶ IPv4 address parameter (Address: 172.16.253.128)
- ▶ Add IP address parameter (Address: 172.16.253.129, correlation ID: 0)



Streaming de video mediante VLC

(API de sockets en C)

Streaming de video mediante VLC

- VLC / HTTP / SCTP:

- Servidor:

- ```
./vlc -vvv --color file.mpg -sout '#standard
{access=http,url=172.16.253.130:8080/file.mpg}'
```

- Cliente:

- ```
./vlc -color http://172.16.253.130:8080/file.mpg
```

- VLC / RTSP / SCTP:

- Servidor

- ```
./vlc -vvv --color -I telnet --rtsp-host
172.16.253.130:5554 --vlm-conf=vlm.conf
```

- Cliente

- ```
./vlc --color rtsp://172.16.253.130:5554/Test
```

Streaming de video mediante VLC

No. ↓	Time	Source	Destination	Protocol	Info
3	7.412309	172.16.253.130	172.16.253.130	SCTP	INIT
4	7.412517	172.16.253.130	172.16.253.130	SCTP	INIT_ACK
5	7.412636	172.16.253.130	172.16.253.130	SCTP	COOKIE_ECHO
6	7.412879	172.16.253.130	172.16.253.130	SCTP	COOKIE_ACK
7	7.454164	172.16.253.130	172.16.253.130	SCTP	DATA
8	7.454246	172.16.253.130	172.16.253.130	SCTP	SACK
9	7.479604	172.16.253.130	172.16.253.130	SCTP	DATA
10	7.558367	172.16.253.130	172.16.253.130	SCTP	DATA
11	7.558443	172.16.253.130	172.16.253.130	SCTP	SACK
12	7.615145	172.16.253.130	172.16.253.130	SCTP	DATA
13	7.623520	172.16.253.130	172.16.253.130	SCTP	DATA
14	7.623586	172.16.253.130	172.16.253.130	SCTP	SACK
15	7.633202	172.16.253.130	172.16.253.130	SCTP	DATA
16	7.733800	172.16.253.130	172.16.253.130	SCTP	DATA
17	7.733921	172.16.253.130	172.16.253.130	SCTP	SACK
18	7.734097	172.16.253.130	172.16.253.130	SCTP	DATA
23	7.770704	172.16.253.130	172.16.253.130	SCTP	DATA
24	7.770777	172.16.253.130	172.16.253.130	SCTP	SACK
25	7.798231	172.16.253.130	172.16.253.130	SCTP	DATA
26	7.829746	172.16.253.130	172.16.253.130	SCTP	SACK
27	7.829862	172.16.253.130	172.16.253.130	SCTP	SHUTDOWN
28	7.830010	172.16.253.130	172.16.253.130	SCTP	SHUTDOWN_ACK
29	7.830501	172.16.253.130	172.16.253.130	SCTP	SHUTDOWN_COMPLETE

Streaming de video mediante VLC

The image displays three Wireshark packet capture windows. The top window shows Frame 7 (120 bytes on wire, 120 bytes captured) with details for Linux cooked capture, Internet Protocol (Src: 172.16.253.130, Dst: 172.16.253.130), and Stream Control Transmission Protocol (Src Port: 1078, Dst Port: 8080). The middle window shows Frame 9 (152 bytes on wire, 152 bytes captured) with details for Linux cooked capture, Internet Protocol (Src: 172.16.253.130, Dst: 172.16.253.130), and Stream Control Transmission Protocol (Src Port: 1078, Dst Port: 8080). The bottom window shows Frame 16 (116 bytes on wire, 116 bytes captured) with details for Linux cooked capture, Internet Protocol (Src: 172.16.253.130, Dst: 172.16.253.130), and Stream Control Transmission Protocol (Src Port: 8080, Dst Port: 1078). The bottom window also shows the raw data (hex and ASCII) for the captured frame, which is an RTP packet (RTSP/1.0 501 Unimplemented Content-Length: 302).

Frame 7 (120 bytes on wire, 120 bytes captured)

- Linux cooked capture
- Internet Protocol, Src: 172.16.253.130 (172.16.253.130), Dst: 172.16.253.130 (172.16.253.130)
- Stream Control Transmission Protocol, Src Port: 1078 (1078), Dst Port: 8080 (8080)
- Source port: 1078

Frame 9 (152 bytes on wire, 152 bytes captured)

- Linux cooked capture
- Internet Protocol, Src: 172.16.253.130 (172.16.253.130), Dst: 172.16.253.130 (172.16.253.130)
- Stream Control Transmission Protocol, Src Port: 1078 (1078), Dst Port: 8080 (8080)
- Data (87 bytes)

Frame 16 (116 bytes on wire, 116 bytes captured)

- Linux cooked capture
- Internet Protocol, Src: 172.16.253.130 (172.16.253.130), Dst: 172.16.253.130 (172.16.253.130)
- Stream Control Transmission Protocol, Src Port: 8080 (8080), Dst Port: 1078 (1078)
- Data (51 bytes)

Raw data (hex and ASCII) for Frame 16:

```

0030  00 03 00 43 50 6e 95 12 00 00 00 00 00 00 00 00  ...CPn...
0040  52 54 53 50 2f 31 2e 30 20 35 30 31 20 55 6e 69  RTSP/1.0 501 Uni
0050  6d 70 6c 65 6d 65 6e 74 65 64 0d 0a 43 6f 6e 74  mplement ed..Cont
0060  65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 30 32 0d  ent-Leng th: 302.
0070  0a 0d 0a 00  ...
  
```

Data (data), 51 bytes P: 155 D: 153 M: 0 Drops: 0

0070 30 38 30 0d 0a 00 00 00 080...

Data (data), 53 bytes P: 155 D: 153 M: 0 Drops: 0

Prueba 2:

Cliente / Servidor en Java

Cliente / Servidor en Java

```
jlopez@servidor-sctp:~/bin - Cliente - Konsole
Sesión Editar Vista Marcadores Preferencias Ayuda
jlopez@servidor-sctp:~/bin> java -cp . -Djava.library.path=. -Xcheck:jni ClienteOne
172.16.253.130 10000

--- CLIENTE SCTP ONE TO ONE STYLE SOCKET ---
- Conectando con : 172.16.253.130:10000
- Enviando 10 mensajes: Cliente->Servidor

-- Socket cliente cerrado --
jlopez@servidor-sctp:~/bin>

jlopez@servidor-sctp:~/bin - Servidor - Konsole
Sesión Editar Vista Marcadores Preferencias Ayuda
jlopez@servidor-sctp:~/bin> java -cp . -Djava.library.path=. -Xcheck:jni ServidorOne
172.16.253.130 10000

--- SERVIDOR SCTP ONE TO ONE STYLE SOCKET ---
Servidor escuchando en : 172.16.253.130:10000
Puerto : 10000

Esperando connect...
--> Recibido SCTPNotificationAssociationChangeCommUp
- Cliente conectado. Datos:
  - ID Asociacion = 3
  - Cliente accesible mediante las direcciones :
    IPv4: /172.16.253.130
    IPv4: /172.16.253.128
    IPv4: /172.16.253.129
  - En el puerto : 1087

--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Datos recibidos : Hola servidor!! :)
--> Recibido SCTPNotificationShutdownEvent

-- Socket servidor cerrado --
jlopez@servidor-sctp:~/bin>
```

Cliente / Servidor en Java

No. ↓	Time	Source	Destination	Protocol	Info
1	0.000000	172.16.253.130	172.16.253.130	SCTP	INIT
2	0.000698	172.16.253.130	172.16.253.130	SCTP	INIT_ACK
3	0.000860	172.16.253.130	172.16.253.130	SCTP	COOKIE_ECHO
4	0.001034	172.16.253.130	172.16.253.130	SCTP	COOKIE_ACK
5	0.009906	172.16.253.130	172.16.253.130	SCTP	DATA
6	0.009985	172.16.253.130	172.16.253.130	SCTP	SACK
7	0.010233	172.16.253.130	172.16.253.130	SCTP	DATA
8	0.010439	172.16.253.130	172.16.253.130	SCTP	DATA
9	0.010489	172.16.253.130	172.16.253.130	SCTP	SACK
10	0.010685	172.16.253.130	172.16.253.130	SCTP	DATA
11	0.010879	172.16.253.130	172.16.253.130	SCTP	DATA
12	0.010928	172.16.253.130	172.16.253.130	SCTP	SACK
13	0.011117	172.16.253.130	172.16.253.130	SCTP	DATA
14	0.011283	172.16.253.130	172.16.253.130	SCTP	DATA
15	0.011332	172.16.253.130	172.16.253.130	SCTP	SACK
16	0.011517	172.16.253.130	172.16.253.130	SCTP	DATA
17	0.011662	172.16.253.130	172.16.253.130	SCTP	DATA
18	0.011710	172.16.253.130	172.16.253.130	SCTP	SACK
19	0.011893	172.16.253.130	172.16.253.130	SCTP	DATA
20	0.211411	172.16.253.130	172.16.253.130	SCTP	SACK
21	0.211533	172.16.253.130	172.16.253.130	SCTP	SHUTDOWN
22	0.211619	172.16.253.130	172.16.253.130	SCTP	SHUTDOWN_ACK
23	0.211674	172.16.253.130	172.16.253.130	SCTP	SHUTDOWN_COMPLETE

▶ Frame 5 (84 bytes on wire, 84 bytes captured)
 ▶ Linux cooked capture
 ▶ Internet Protocol, Src: 172.16.253.130 (172.16.253.130), Dst: 172.16.253.130 (172.16.253.130)
 ▶ Stream Control Transmission Protocol, Src Port: 1091 (1091), Dst Port: 10000 (10000)
 Data (20 bytes)

```

0000  00 00 03 04 00 00 00 00 00 00 00 00 00 00 08 00  .....
0010  45 02 00 44 00 01 40 00 40 84 e7 0c ac 10 fd 82  E..D..@. @.....
0020  ac 10 fd 82 04 43 27 10 d3 8b 08 5a 00 00 00 00  ...C'. ...Z....
0030  00 03 00 24 d4 6c eb 55 00 00 00 00 00 00 00 00  ...$.L.U .....
0040  09 20 48 6f 6c 61 20 73 65 72 76 69 64 6f 72 21  . Hola s ervidor!
0050  21 20 3a 25                                     ! :)
  
```



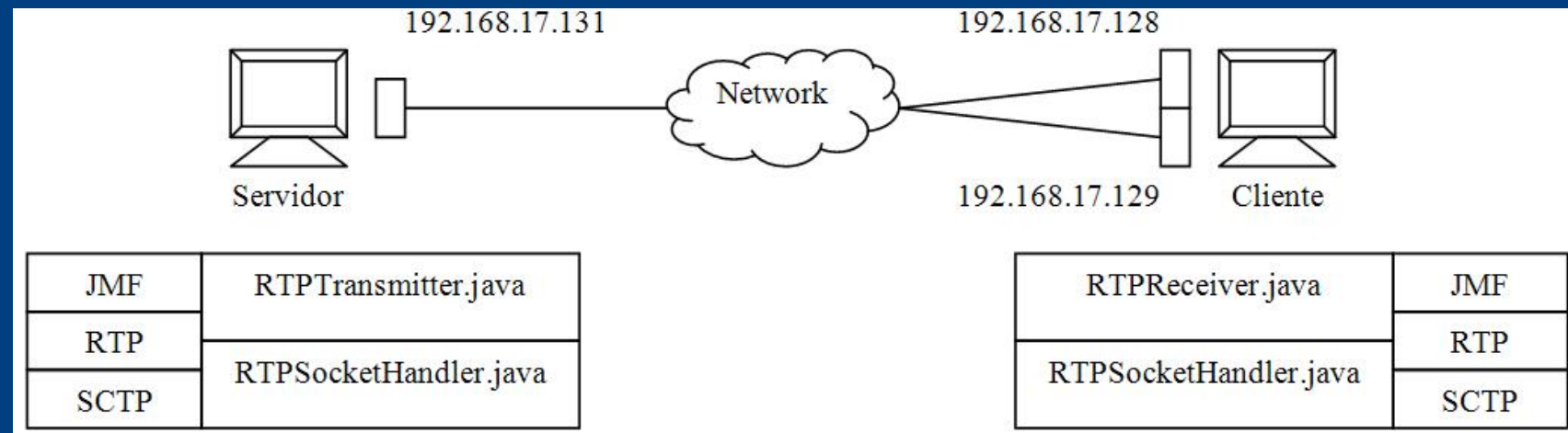
Streaming de video mediante JMF

(API de sockets en Java)

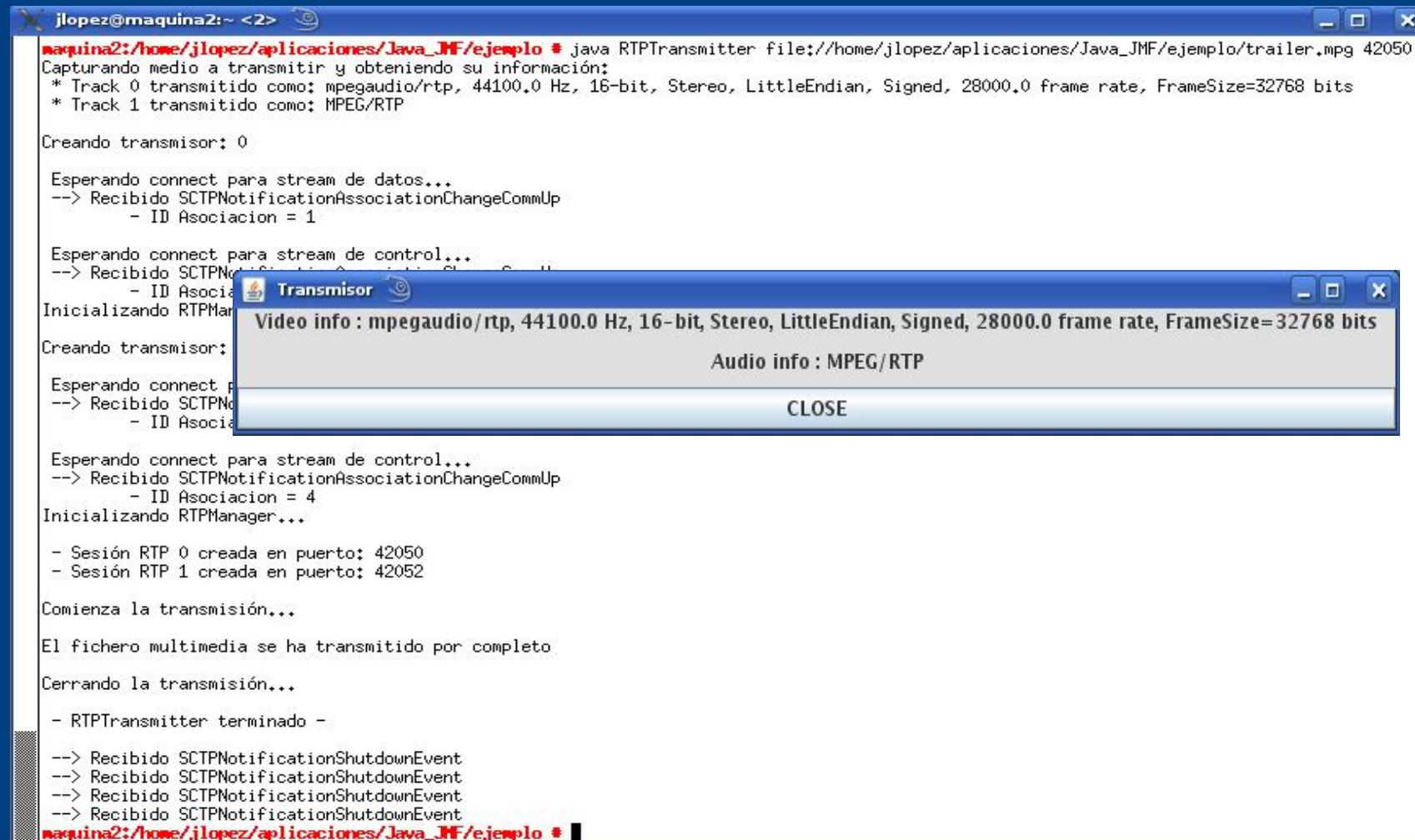
Streaming de video mediante JMF

Java Media Framework (JMF) es un API que proporciona herramientas para la captura, procesamiento, almacenamiento y reproducción de datos multimedia.

Prueba. Escenario:



Streaming de video mediante JMF



```
jlopez@maquina2: ~ <2>
maquina2:/home/jlopez/aplicaciones/Java_JMF/ejemplo # java RTPTransmitter file://home/jlopez/aplicaciones/Java_JMF/ejemplo/trailer.mpg 42050
Capturando medio a transmitir y obteniendo su información:
* Track 0 transmitido como: mpegaudio/rtp, 44100.0 Hz, 16-bit, Stereo, LittleEndian, Signed, 28000.0 frame rate, FrameSize=32768 bits
* Track 1 transmitido como: MPEG/RTP

Creando transmisor: 0

Esperando connect para stream de datos...
--> Recibido SCTPNotificationAssociationChangeCommUp
    - ID Asociacion = 1

Esperando connect para stream de control...
--> Recibido SCTPNotificationAssociationChangeCommUp
    - ID Asociacion = 1
Iniciando RTPManager...
Creando transmisor: 0

Esperando connect para stream de datos...
--> Recibido SCTPNotificationAssociationChangeCommUp
    - ID Asociacion = 4
Iniciando RTPManager...

- Sesión RTP 0 creada en puerto: 42050
- Sesión RTP 1 creada en puerto: 42052

Comienza la transmisión...

El fichero multimedia se ha transmitido por completo

Cerrando la transmisión...

- RTPTransmitter terminado -

--> Recibido SCTPNotificationShutdownEvent
--> Recibido SCTPNotificationShutdownEvent
--> Recibido SCTPNotificationShutdownEvent
--> Recibido SCTPNotificationShutdownEvent
maquina2:/home/jlopez/aplicaciones/Java_JMF/ejemplo #
```

Transmisor

Video info : mpegaudio/rtp, 44100.0 Hz, 16-bit, Stereo, LittleEndian, Signed, 28000.0 frame rate, FrameSize=32768 bits

Audio info : MPEG/RTP

CLOSE

Streaming de video mediante JMF

```
jlopez@maquina1:~  
maquina1:/home/jlopez/aplicaciones/Java_JMF/ejemplo # java RTPReceiver 192.168.17.131/42050 192.168.17.131/42052  
- Sesión RTP abierta --> IP: 192.168.17.131 Puerto: 42050  
  - Conectando con stream de datos en 192.168.17.131:42050  
--> Recibido SCTPNotificationAssociationChangeCommUp  
  - ID Asociacion = 1  
  - Conectando con stream de control en 192.168.17.131:42051  
--> Recibido SCTPNotificationAssociationChangeCommUp  
  - ID Asociacion = 2  
Iniciando RTPManager...  
  
- Sesión RTP abierta --> IP: 192.168.17.131 Puerto: 42052  
  - Conectando con stream de datos en 192.168.17.131:42052  
--> Recibido SCTPNotificationAssociationChangeCommUp  
  - ID Asociacion = 3  
  - Conectando con stream de control en 192.168.17.131:42053  
--> Recibido SCTPNotificationAssociationChangeCommUp  
  - ID Asociacion = 4  
- Un nuevo participante se acaba de unir: root@maquina2  
Iniciando RTPManager...  
  
- Esperando la llegada de datos RTP...  
- Esperando la llegada de datos RTP...  
- Un nuevo participante se acaba de unir: root@maquina2  
- Esperando la llegada de datos RTP...  
- Recibido un nuevo Stream RTP: mpegaudio/rtp, Unknown Sample Rate  
  El stream proviene de: root@maquina2  
- Esperando la llegada de datos RTP...  
- Recibido un nuevo Stream RTP: MPEG/RTP  
  El stream proviene de: root@maquina2  
- Recibido bye desde: root@maquina2  
  
Cerrando recepción...  
  
- RTPReceiver terminado -  
  
maquina1:/home/jlopez/aplicaciones/Java_JMF/ejemplo #
```



Streaming de video mediante JMF

No. ↓	Time	Source	Destination	Protocol	Info
5	52.462749	192.168.17.129	192.168.17.131	SCTP	INIT
10	52.467134	192.168.17.131	192.168.17.129	SCTP	INIT_ACK
11	52.467221	192.168.17.129	192.168.17.131	SCTP	COOKIE_ECHO
12	52.467298	192.168.17.131	192.168.17.129	SCTP	COOKIE_ACK
13	53.481444	192.168.17.129	192.168.17.131	SCTP	INIT
14	53.482213	192.168.17.131	192.168.17.129	SCTP	INIT_ACK
15	53.482272	192.168.17.129	192.168.17.131	SCTP	COOKIE_ECHO
16	53.482765	192.168.17.131	192.168.17.129	SCTP	COOKIE_ACK
17	54.056167	192.168.17.129	192.168.17.131	SCTP	DATA
18	54.056616	192.168.17.131	192.168.17.129	SCTP	SACK
19	54.056643	192.168.17.129	192.168.17.131	SCTP	DATA
20	54.056852	192.168.17.131	192.168.17.129	SCTP	SACK
21	55.069949	192.168.17.129	192.168.17.131	SCTP	DATA
22	55.070672	192.168.17.131	192.168.17.129	SCTP	SACK
23	55.070732	192.168.17.129	192.168.17.131	SCTP	DATA
24	55.071230	192.168.17.131	192.168.17.129	SCTP	SACK

No. ↓	Time	Source	Destination	Protocol	Info
32235	165.447324	192.168.17.129	192.168.17.131	SCTP	SACK
32236	165.486667	192.168.17.131	192.168.17.129	SCTP	DATA
32237	165.486726	192.168.17.129	192.168.17.131	SCTP	SACK
32238	165.486958	192.168.17.131	192.168.17.129	SCTP	DATA
32239	165.695208	192.168.17.129	192.168.17.131	SCTP	SACK
32240	165.695244	192.168.17.129	192.168.17.131	SCTP	SACK
32241	165.696620	192.168.17.131	192.168.17.129	SCTP	DATA
32242	166.561757	192.168.17.131	192.168.17.128	SCTP	DATA

No. ↓	Time	Source	Destination	Protocol	Info
41987	175.699915	192.168.17.128	192.168.17.131	SCTP	DATA
41988	175.898594	192.168.17.131	192.168.17.128	SCTP	SACK
41989	176.591479	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN
41990	176.591991	192.168.17.131	192.168.17.128	SCTP	SHUTDOWN_ACK
41991	176.592683	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN_COMPLETE
41992	176.644903	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN
41993	176.645467	192.168.17.131	192.168.17.128	SCTP	SHUTDOWN_ACK
41994	176.645921	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN_COMPLETE
41995	176.645972	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN
41996	176.646046	192.168.17.131	192.168.17.128	SCTP	SHUTDOWN_ACK
41997	176.646069	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN_COMPLETE
41998	176.646338	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN
41999	176.646652	192.168.17.131	192.168.17.128	SCTP	SHUTDOWN_ACK
42000	176.646678	192.168.17.128	192.168.17.131	SCTP	SHUTDOWN_COMPLETE

4. Conclusiones y trabajos futuros

Conclusiones

- API de sockets en C completo, ofrece funcionalidades esperadas.
- API de sockets en Java bastante limitado.
- Streaming de video mediante VLC no conseguido, código complejo, difícil de abordar.
- Uso de JMF satisfactorio, permitió realizar streaming de video sobre SCTP.

Trabajos futuros

- Finalizar el API de sockets en Java o probar alguno más completo.
- Finalizar la prueba de streaming de video mediante VLC.
- Mejorar la aplicación realizada en JMF: más características de reproductor de video y API de sockets en Java más completo.
- Introducir SCTP en plataformas de VoIP.
- Portar este proyecto a plataforma windows para realizar una comparativa.

5. Bibliografía

- KNF: “Stream Control Transmission Protocol (SCTP) – RFCs related to SCTP”. [On line]. Disponible en WWW: <http://www.sctp.de/sctp.html>.
- Craig Rodrigues: “Stream Control Transmission Protocol (SCTP). [On line]. Disponible en WWW: <http://www.sctp.org>. Octubre 2006.
- SourceForge: “Linux Kernel SCTP”. [On line]. Disponible en WWW: <http://sourceforge.net/projects/lksctp/>. Agosto 2008.
- VMware, Inc: “VMware Player”. [On line]. Disponible en WWW: <http://www.vmware.com/products/player/>. 2008.
- Ivan Skytte Jorgensen: “Java SCTP Library – SCTP socket interface for Java (using JNI)”. [On line]. Disponible en WWW: <http://i1.dk/JavaSCTP/>.
- VideoLAN: “VLC media player for OpenSUSE x86 and x86_64”. [On line]. Disponible en WWW: <http://www.videolan.org/vlc/download-suse.html>. 2008.
- Sun Microsystems: “Java SE Desktop Technologies”. [On line]. Disponible en WWW: <http://java.sun.com/javase/technologies/desktop/media/jmf/>

Fin

GRACIAS POR SU ATENCIÓN

